**Learners have to come up with a Report to support the answers to the following questions and suggestions**

Objective Questions

1. Are there any tables with duplicate or missing null values? If so, how would you handle them?

Answer- There are no Null or Duplicate rows in the data.

Example of query used for this task-

**Null-**

**SELECT \* FROM tags**

**WHERE id IS NULL**

**OR tag\_name IS NULL**

**OR created\_at IS NULL;**

**Duplicate-**

**SELECT id, image\_url, user\_id, created\_dat, COUNT(\*)**

**FROM photos**

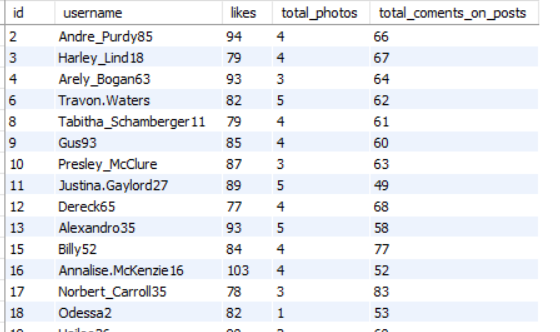
**GROUP BY id, image\_url, user\_id, created\_dat**

**HAVING COUNT(\*) > 1;**

-------------------------------------------------------

1. What is the distribution of user activity levels (e.g., number of posts, likes, comments) across the user base?

Answer



with comments\_per\_users as (

select user\_id, count(id) as total\_coments\_on\_posts

from comments

group by 1

),

likes\_per\_users as (

select user\_id, count(photo\_id) as likes

from likes

group by 1

),

posts\_per\_users as (

select user\_id,count(id) as total\_photos

from photos

group by 1

)

select id, username, likes,total\_photos,total\_coments\_on\_posts

from

users u join likes\_per\_users l

on u.id = l.user\_id

join comments\_per\_users c on u.id = c.user\_id

join posts\_per\_users p on u.id = p.user\_id

order by id;

**Explanation**- This query outputs the total number of likes, photo posts, comments done by an user. In addition, the user activity level is the sum of likes, photo posts, comments done by the respective user.

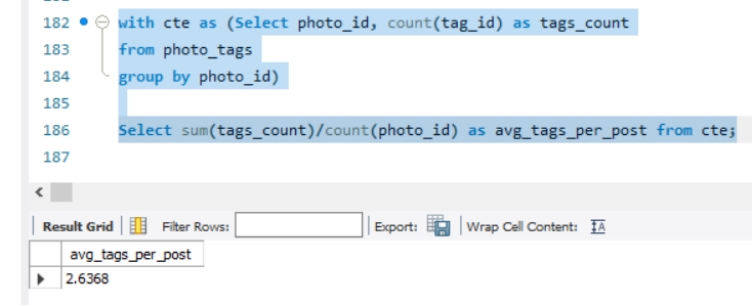
1. Calculate the average number of tags per post (photo\_tags and photos tables).

with cte as (Select photo\_id, count(tag\_id) as tags\_count

from photo\_tags

group by photo\_id)

Select sum(tags\_count)/count(photo\_id) as avg\_tags\_per\_post from cte;



-----------------------------

1. Identify the top users with the highest engagement rates (likes, comments) on their posts and rank them.

Answer-

with comments\_per\_users as (

select user\_id, count(id) as total\_coments\_on\_posts

from comments

group by 1

),

likes\_per\_users as (

select user\_id, count(photo\_id) as likes

from likes

group by 1

),

user\_engagement as (select id, username, sum(total\_coments\_on\_posts + likes) as engagement\_rate

from users u

join likes\_per\_users l

on u.id = l.user\_id

join comments\_per\_users c on u.id = c.user\_id

group by 1

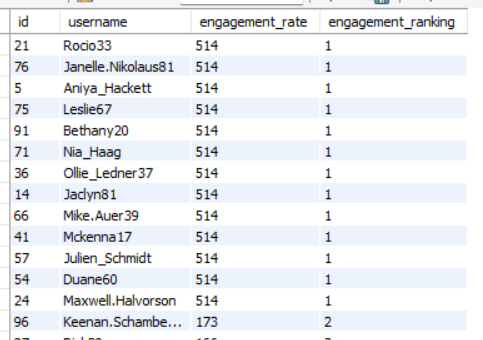
)

select \*, dense\_rank() over(order by engagement\_rate desc) as engagement\_ranking

from

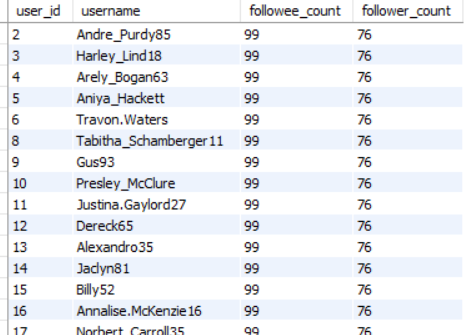
user\_engagement

order by engagement\_ranking;



Explanation- The first CTE (comments\_per\_user) counts the total number of comments done by a user, the second CTE (likes\_per\_users) counts the number of likes done by an user. And the third CTE (user\_engagement) which uses the above two CTEs counts the total number of comments and likes. Finally the last select statement ranks these users based on their sum of likes and counts.

1. Which users have the highest number of followers and followings?



with followee\_cte as (Select u.id, u.username, count(f.followee\_id) followee\_count

from users u join follows f

on u.id=f.follower\_id

group by u.id),

follower\_cte as (select u.id, u.username, count(f.follower\_id) as follower\_count

from users u join follows f

on u.id=f.followee\_id

group by u.id)

select r.id as user\_id, r.username, e.followee\_count, r.follower\_count

from followee\_cte e join follower\_cte r

on r.id=e.id

order by r.follower\_count desc, e.followee\_count asc;

**Explanation**- The first CTE (followee\_cte) counts the number of followee for each user. The second CTE (follower\_cte) counts the number of followers for each user. The third and final select statement, represents the number of followers, and followees for each user.

1. Calculate the average engagement rate (likes, comments) per post for each user.

with avg\_comment as (select user\_id, username, round(sum(comment\_count)/count(photo\_id),1) as average\_comment\_count

from (Select u.id as user\_id, u.username, p.id as photo\_id, count(c.id) as comment\_count

from users u

left join photos p

on u.id=p.user\_id

join comments c

on p.id=c.photo\_id

group by u.id, p.id) alias1

group by user\_id, username),

avg\_like as (select user\_id, round(sum(like\_count)/count(photo\_id),1) as average\_like\_count

from (Select u.id as user\_id, p.id as photo\_id, count(l.photo\_id) as like\_count

from users u

right join photos p

on u.id=p.user\_id

join likes l

on p.id=l.photo\_id

group by u.id, p.id) alias2

group by user\_id)

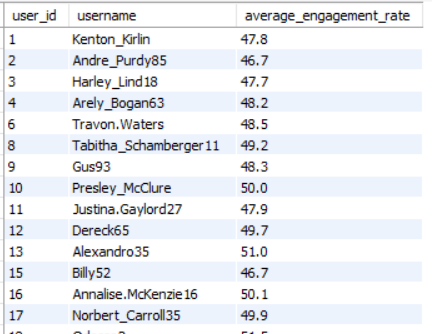
select c.user\_id, c.username, round(l.average\_like\_count+c.average\_comment\_count/2,1)

as average\_engagement\_rate

from avg\_comment c join avg\_like l

on c.user\_id=l.user\_id

group by c.user\_id, c.username;

**

**Explaination**- In the subquery of the first CTE (‘avg\_comment’) i determined the average number of comments per post for each user using group by and aggregate functions. In main query of the same CTE I determined the average count of comments for each post per user. Furthermore, I used the same approach in the second CTE (avg\_like) for determining the average count of likes for each post per user.

Later, I combined both the CTEs in which I added the average count of comments and likes and devided by 2 to get the average engagement rate for each user.

1. Get the list of users who have never liked any post (users and likes tables)

Answer-

with user\_likes as (

Select u.id, u.username, count(l.photo\_id) as likes\_count

from users u

left join likes l

on l.user\_id=u.id

group by u.id

)

Select id, username from user\_likes

where likes\_count = 0;



1. How can you leverage user-generated content (posts, hashtags, photo tags) to create more personalized and engaging ad campaigns?

Answer- The social media company can identify the users with high number of likes and comments and show ads on them.

with tagcount as (

Select l.user\_id, p.photo\_id, t.tag\_name,

count(tag\_name) over(partition by tag\_name order by user\_id) as count\_tag

from likes l join photo\_tags p

on l.photo\_id=p.photo\_id

left join tags t

on p.tag\_id=t.id

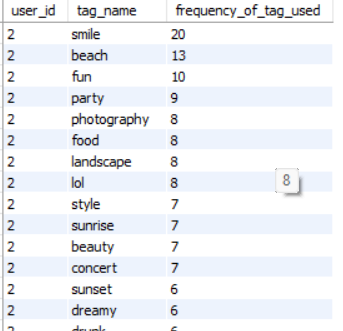
order by user\_id)

Select user\_id, tag\_name, count(count\_tag) as frequency\_of\_tag\_used

from tagcount

group by user\_id, tag\_name

order by user\_id, count(count\_tag) desc;



Explanation- In the CTE (tagcount) I determined the name and count of tags (using window function) per photo for each user. Later in the query I outputted the user id, tag name, and the count of a particular tag used by a specific user. The final output is sorted based on user id, and count of tags in descending order for each user.

The above table represents all the tags and its frequency of use by a user. The social media company can use this data to target their campaign based on the type of tags a particular user have used most of the times.

1. Are there any correlations between user activity levels and specific content types (e.g., photos, videos, reels)? How can this information guide content creation and curation strategies?

Answer-

with likes\_per\_users as (

select

user\_id, count(photo\_id) as num\_likes

from

likes

group by 1

),

comments\_per\_posts as (

select

user\_id, count(comment\_text) as num\_coments\_on\_posts

from

comments

group by 1

),

posts\_per\_users as (

select

user\_id, count(image\_url) as posts

from

photos

group by 1

),

engagement\_per\_user as (

select

username as users\_name,

p.user\_id,

p.id,

sum(num\_likes + num\_coments\_on\_posts + posts) as total\_engagement\_rate

from

photos p

join likes\_per\_users c1 on p.user\_id = c1.user\_id

join comments\_per\_posts c2 on p.user\_id = c2.user\_id

join users u on p.user\_id = u.id

join posts\_per\_users c3 on p.user\_id = c3.user\_id

group by 1, 2, 3

),

engagement\_percentage as (

select

users\_name,

sum(round(100 \* (num\_likes / total\_engagement\_rate), 2)) as num\_likes\_percent,

sum(round(100 \* (posts / total\_engagement\_rate), 2)) as posts\_percent,

sum(round(100 \* (num\_coments\_on\_posts / total\_engagement\_rate), 2)) as comment\_percent,

sum(round(100 \* (total\_engagement\_rate / total\_engagement\_rate), 2)) as total\_engagement\_percent

from

engagement\_per\_user c4

join likes\_per\_users c1 on c4.user\_id = c1.user\_id

join comments\_per\_posts c2 on c4.user\_id = c2.user\_id

join users u on c4.user\_id = u.id

join posts\_per\_users c3 on c4.user\_id = c3.user\_id

group by 1

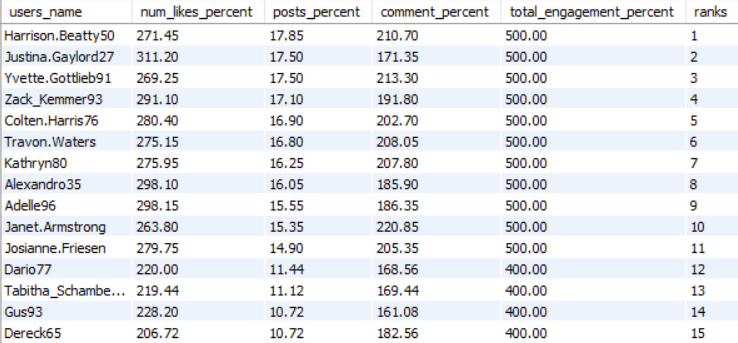
)

select \*, dense\_rank() over (order by total\_engagement\_percent desc, posts\_percent desc, num\_likes\_percent desc) as ranks

from

engagement\_percentage

order by ranks;



Explanation-

The analysis involved summing all three activity count columns in a fourth CTE to derive user activity levels. Next, the number of likes was divided by the total engagement rate to calculate the percentage of likes per user. A similar method was applied to determine both the post percentage and the comment percentage. The dense\_rank function was then utilized to distinguish users with the highest engagement rates. The output indicated that users with higher engagement rates tended to post more photos compared to those with lower engagement rates. This suggests a strong correlation between user activity levels and specific content types.

1. Calculate the total number of likes, comments, and photo tags for each user.

with comments\_per\_users as (

select c.user\_id, u.username, count(c.id) as total\_coments

from comments c join users u

on c.user\_id=u.id

group by c.user\_id, u.username

),

likes\_per\_users as (

select l.user\_id, u.username, count(photo\_id) as likes

from likes l join users u

on l.user\_id=u.id

group by user\_id,u.username

),

tags\_per\_users as (

Select u.id as user\_id, u.username, count(pt.tag\_id) as tag\_count

from users u

right join photos p

on u.id=p.user\_id

join photo\_tags pt

on p.id=pt.photo\_id

group by u.id, u.username

)

select u.id, u.username, l.likes, c.total\_coments, t.tag\_count

from

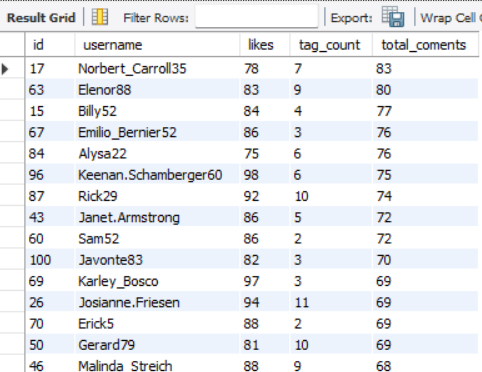
users u join likes\_per\_users l

on u.id = l.user\_id

join comments\_per\_users c on u.id = c.user\_id

join tags\_per\_users t on u.id = t.user\_id

order by c.total\_coments desc, l.likes desc, t.tag\_count desc;



Explanation-

I created three CTEs to count the number of likes, comments, and photo tags. Then, I grouped the data by user\_id to calculate the total count of likes, comments, and tags for each user.

1. Rank users based on their total engagement (likes, comments, shares) over a month.

Answer-

with likes\_per\_posts as (

select

id, count(photo\_id) as num\_likes

from

likes l

join

photos p on l.photo\_id = p.id

group by 1

),

comments\_per\_post as (

select

photo\_id, count(comment\_text) as num\_coments\_on\_posts

from

comments

group by 1

),

per\_month\_engagement\_ranking as (

select

username as users\_name,

month(created\_at) as months,

sum(num\_likes + num\_coments\_on\_posts) as total\_engagement\_rate,

dense\_rank() over(partition by month(created\_at)

order by sum(num\_likes + num\_coments\_on\_posts) desc) month\_wise\_engagement\_rank

from

photos p

join

likes\_per\_posts c1 on p.id = c1.id

join

comments\_per\_post c2 on p.id = c2.photo\_id

join

users u on p.user\_id = u.id

group by 1, month(created\_at)

)

select

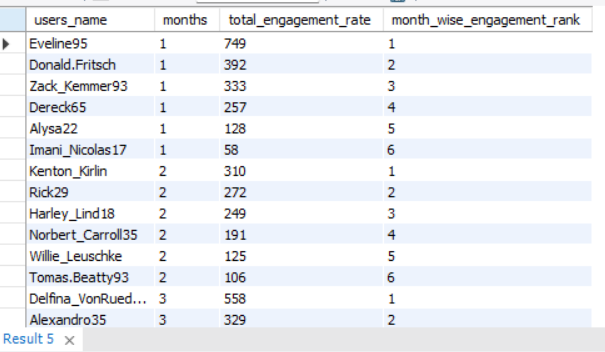
\*

from

per\_month\_engagement\_ranking

order by 2;

Explaination- In this query, I calculated and ranked user engagement on posts by month. First, I counted the number of likes per post by joining the likes and photos tables. Then, I counted the number of comments on each post. After that, I combined the likes and comments data to calculate the total engagement for each post. I grouped the results by username and month, then assigned a rank to each user based on their total engagement for that month. Finally, I selected and displayed the monthly engagement rankings, ordered by month*.*

**

1. Retrieve the hashtags that have been used in posts with the highest average number of likes. Use a CTE to calculate the average likes for each hashtag first.

Answer-

with likes\_per\_photo as (

select

photo\_id, count(user\_id) as num\_likes

from

likes

group by 1

),

avg\_likes\_per\_tag\_rank as (

select

tag\_name, round(avg(num\_likes), 2) as avg\_num\_likes,

dense\_rank() over(order by avg(num\_likes) desc) as tag\_avg\_likes\_rank

from

likes\_per\_photo c1

join

photo\_tags p on c1.photo\_id = p.photo\_id

join

tags t on t.id = p.tag\_id

group by 1

)

select

upper(tag\_name) as max\_liked\_tag,

avg\_num\_likes as max\_avg\_num\_likes,

tag\_avg\_likes\_rank

from

avg\_likes\_per\_tag\_rank

where

tag\_avg\_likes\_rank = 1

order by 2;



Explaination- In this query, I retrieved the hashtags used in posts with the highest average number of likes. First, I used a CTE (Common Table Expression) to calculate the number of likes for each photo by grouping the likes data. Then, I calculated the average number of likes for each hashtag by joining the likes data with the photo tags and tags tables. I ranked the hashtags based on their average likes using the dense\_rank() function. Finally, I selected and displayed the hashtag with the highest average number of likes, converting the tag name to uppercase and ordering the results by the average number of likes.

1. Retrieve the users who have started following someone after being followed by that person

with Users\_followed\_after\_being\_followed as (

SELECT DISTINCT fe.follower\_id AS user\_who\_followed\_after, fe.followee\_id AS followed\_by

FROM

follows fe

JOIN

follows fo ON fe.followee\_id = fo.follower\_id

AND fe.follower\_id != fo.follower\_id

WHERE

fe.follower\_id = fo.followee\_id

)

SELECT DISTINCT username AS Later\_followed\_users

FROM

Users\_followed\_after\_being\_followed c1

JOIN users u ON c1.user\_who\_followed\_after = u.id

ORDER BY 1;



In this query, I retrieved users who started following someone after being followed by that person. I used a CTE called Users\_followed\_after\_being\_followed to identify cases where a user followed someone after being followed by them. This was achieved by joining the follows table on the follower and followee relationship, ensuring that the follower IDs were different but matched in reverse order. Finally, I selected and displayed the usernames of users who followed back, ensuring no duplicates, and ordered the results alphabetically.

Subjective Questions

1. Based on user engagement and activity levels, which users would you consider the most loyal or valuable? How would you reward or incentivize these users?

Answer-

with comments\_per\_users as (

select user\_id, count(id) as total\_coments\_per\_user

from comments

group by 1

),

likes\_per\_users as (

select user\_id, count(photo\_id) as likes

from likes

group by 1

),

posts\_per\_user as (

Select user\_id, count(image\_url) as total\_posts

From photos

Group by 1

),

user\_engagement as (select id, username, sum(total\_coments\_per\_user + likes) as engagement\_rate,

sum(total\_posts + total\_coments\_per\_user + likes) as activity\_level\_users

from users u

join likes\_per\_users l

on u.id = l.user\_id

join comments\_per\_users c on u.id = c.user\_id

join posts\_per\_user p on u.id=p.user\_id

group by 1

)

select \*, dense\_rank() over(order by activity\_level\_users desc, engagement\_rate) as user\_ranking

from user\_engagement

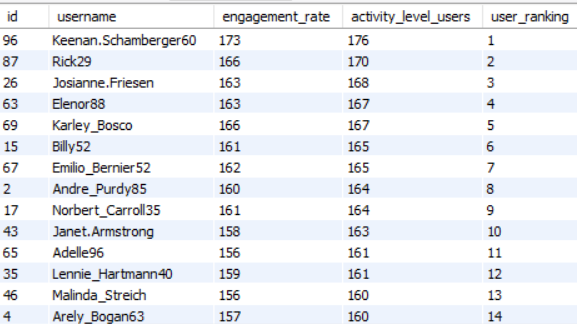
where activity\_level\_users > (select avg(activity\_level\_users)

from user\_engagement) and engagement\_rate > (select

avg(engagement\_rate)

from user\_engagement)

Order by 5;



In this query, I calculated and ranked user engagement and activity levels. I started by creating three CTEs to count user interactions: the first CTE calculates the total comments per user, the second counts likes per user, and the third counts total posts per user. I then combined these in a fourth CTE called user\_engagement, where I calculated each user's engagement rate (sum of comments and likes) and activity level (sum of posts, comments, and likes). Finally, I ranked users based on their activity level and engagement rate, filtering only those above the average for both metrics. The final result displays the ranking and orders the users by activity level and engagement.

1. For inactive users, what strategies would you recommend to re-engage them and encourage them to start posting or engaging again?

Answer-

with comments\_per\_users as (

select user\_id, count(id) as total\_coments\_per\_user

from comments

group by 1

),

likes\_per\_users as (

select user\_id, count(photo\_id) as likes

from likes

group by 1

),

posts\_per\_user as (

Select user\_id, count(image\_url) as total\_posts

From photos

Group by 1

),

user\_engagement as (select id, username, sum(total\_coments\_per\_user + likes) as engagement\_rate,

sum(total\_posts + total\_coments\_per\_user + likes) as activity\_level\_users

from users u

left join likes\_per\_users l

on u.id = l.user\_id

left join comments\_per\_users c on u.id = c.user\_id

left join posts\_per\_user p on u.id=p.user\_id

where l.user\_id is null

group by 1

)

select username as Inactive\_users

from user\_engagement

order by 1;



In this query, I retrieved a list of inactive users based on their engagement and activity levels. First, I created three CTEs: the first one counts the total comments per user, the second counts likes per user, and the third counts total posts per user. I then combined these in a fourth CTE called user\_engagement, where I calculated each user's engagement rate and activity level. The key part is that I used LEFT JOIN to include users even if they haven't liked, commented, or posted anything, filtering for users where likes\_per\_users is null. Finally, I selected and displayed the usernames of these inactive users, ordered alphabetically.

1. Which hashtags or content topics have the highest engagement rates? How can this information guide content strategy and ad campaigns?

Answer-

with avg\_like\_count\_cte as (select avg(like\_count) as avg\_likes from

(Select p.id as photo\_id, count(l.created\_at) as like\_count

from photos p

join likes l

on p.id=l.photo\_id

group by p.id) alias)

select tag\_name, like\_count from (select distinct tag\_name, like\_count,

row\_number() over(partition by tag\_name order by like\_count desc) as rn

from (select l.photo\_id, t.tag\_name, count(l.created\_at) as like\_count

from likes l

join photo\_tags pt

on l.photo\_id=pt.photo\_id

join tags t

on pt.tag\_id=t.id

group by l.photo\_id, t.tag\_name

having count(l.created\_at) > (select avg\_likes from avg\_like\_count\_cte)

order by like\_count desc) alias) alias

where rn=1

order by like\_count desc



In this query, I identified hashtags or content topics with the highest engagement rates by focusing on likes. First, I created a CTE (avg\_like\_count\_cte) to calculate the average number of likes per post. Then, I retrieved the hashtags that are associated with posts receiving more likes than the average, using joins between the likes, photo\_tags, and tags tables. I used row\_number() to rank the hashtags by like count and filtered the top-performing hashtags. Finally, I selected and displayed the hashtags with the highest engagement rates, ordered by the number of likes.

This information can guide content strategy and ad campaigns by highlighting the most engaging topics, allowing for targeted promotions and content that resonates with users based on historical data.

1. Are there any patterns or trends in user engagement based on demographics (age, location, gender) or posting times? How can these insights inform targeted marketing campaigns?

Answer-

with time\_of\_likes as (

SELECT

photo\_id,

TIME(created\_at) AS likes\_time,

COUNT(photo\_id) AS num\_likes

FROM

likes

GROUP BY 1 , 2

),

time\_of\_comments as (

SELECT

photo\_id,

TIME(created\_at) AS comments\_time,

COUNT(comment\_text) AS num\_coments\_per\_users

FROM

comments

GROUP BY 1 , 2

),

time\_based\_engagement as (

SELECT

TIME(created\_dat) AS post\_time,

comments\_time,

likes\_time,

SUM(num\_coments\_per\_users + num\_likes) AS Engagement\_rate

FROM

photos p

JOIN

time\_of\_likes c1 ON p.id = c1.photo\_id

JOIN

time\_of\_comments c2 ON p.id = c2.photo\_id

GROUP BY 1 , 2 , 3

)

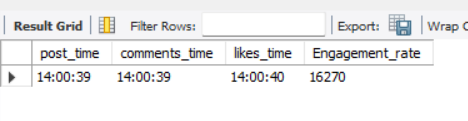
SELECT

\*

FROM

time\_based\_engagement

ORDER BY 1;



Explaination-

This SQL query analyzes user engagement on photos by examining the timing of likes and comments. It first creates two subqueries: time\_of\_likes and time\_of\_comments. In these, it groups likes and comments by photo ID and the exact time (TIME(created\_at)) they were created. The time\_of\_likes subquery counts the likes per photo at specific times, while time\_of\_comments counts the number of comments per photo during those times.

Next, the query combines these two results into a new table time\_based\_engagement, where it joins the photos table with the time\_of\_likes and time\_of\_comments tables on the photo ID. It then calculates the overall engagement rate (sum of likes and comments) for each photo by the time they were posted, and the time of corresponding likes and comments. The final result is sorted by the post time (post\_time), providing a time-based analysis of engagement for photos.

This approach helps understand how engagement (likes and comments) fluctuates based on the time of posting and interaction.

1. Based on follower counts and engagement rates, which users would be ideal candidates for influencer marketing campaigns? How would you approach and collaborate with these influencers?

Answer

with Users\_follow\_count as (

SELECT

followee\_id, COUNT(DISTINCT follower\_id) AS Follower\_count

FROM

follows

GROUP BY 1

),

like\_per\_user as (

SELECT

user\_id, COUNT(photo\_id) AS num\_likes

FROM

likes

GROUP BY 1

),

comments\_per\_user as (

SELECT

user\_id, COUNT(comment\_text) AS num\_coments\_per\_users

FROM

comments

GROUP BY 1

),

Ranked\_Users\_for\_marketing as (

SELECT

username AS Platform\_Username,

Follower\_count,

SUM(num\_coments\_per\_users + num\_likes) AS Engagement\_rate,

DENSE\_RANK() OVER(ORDER BY SUM(num\_coments\_per\_users + num\_likes) DESC, follower\_count DESC) as Rankings

FROM

users u

JOIN

Users\_follow\_count c1 ON u.id = c1.followee\_id

JOIN

like\_per\_user l ON u.id = l.user\_id

JOIN

comments\_per\_user c2 ON u.id = c2.user\_id

GROUP BY 1 , 2

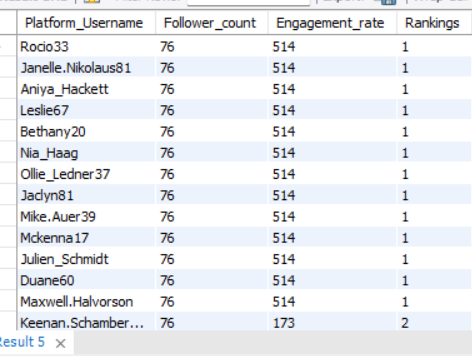
)

SELECT

\*

FROM

Ranked\_Users\_for\_marketing;

**

1. Based on user behavior and engagement data, how would you segment the user base for targeted marketing campaigns or personalized recommendations?

Answer-

with like\_per\_user as (

SELECT

user\_id, COUNT(photo\_id) AS num\_likes

FROM

likes

GROUP BY 1

),

comments\_per\_user as (

SELECT

user\_id, COUNT(comment\_text) AS num\_coments\_per\_users

FROM

comments

GROUP BY 1

),

Tags\_per\_user as (

SELECT

tag\_name, user\_id

FROM

tags t

JOIN

photo\_tags pt ON t.id = pt.tag\_id

JOIN

photos p ON p.id = pt.photo\_id

),

Ranked\_Users\_for\_marketing as (

SELECT

username AS Platform\_Username,

tag\_name as Users\_Tags,

SUM(num\_coments\_per\_users + num\_likes) AS Per\_Tag\_Engagement\_rate,

dense\_rank() OVER(PARTITION BY username ORDER BY SUM(num\_coments\_per\_users + num\_likes) DESC) AS Users\_Per\_tag\_ranking

FROM

users u

JOIN

like\_per\_user l ON u.id = l.user\_id

JOIN

comments\_per\_user c2 ON u.id = c2.user\_id

join

Tags\_per\_user t on u.id = t.user\_id

GROUP BY 1 , 2

)

SELECT

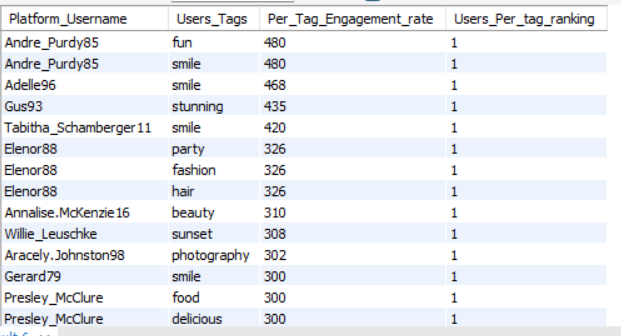
\*

FROM

Ranked\_Users\_for\_marketing

where Users\_Per\_tag\_ranking = 1

ORDER BY 3 desc,1;



This query identifies and ranks users based on their engagement (likes and comments) across specific tags for marketing purposes. First, it calculates the total number of likes and comments per user in the like\_per\_user and comments\_per\_user subqueries. Then, the Tags\_per\_user subquery associates each user with the tags they’ve used by joining the tags, photo tags, and photos tables. In the Ranked\_Users\_for\_marketing step, it sums the likes and comments for each user and tag to calculate the "Per\_Tag\_Engagement\_rate" and ranks users per tag using DENSE\_RANK(). Finally, it filters the result to show only the top-ranked users for each tag, ordering them by engagement rate for marketing insights.

1. If data on ad campaigns (impressions, clicks, conversions) is available, how would you measure their effectiveness and optimize future campaigns?

Answer- To measure and optimize the effectiveness of ad campaigns using data on impressions, clicks, conversions, and user-liked tags, start by defining KPIs such as CTR (click-through rate) and conversion rate. Segment the data by user-liked tags to analyze the performance of each tag. Conduct A/B testing to compare different ad variations targeting various tags and assess their effectiveness. Use the insights from these tests to optimize your targeting by focusing on high-performing tags. Refine your ad creatives to highlight popular tags and test different messages to find what resonates best with your audience. Allocate your budget more effectively by increasing spending on tags that drive higher engagement and conversions, while reducing it for underperforming tags. Continuously monitor the performance metrics and make data-driven adjustments to your campaigns. Regularly summarize insights and key takeaways to inform and improve future ad strategies.

1. How can you use user activity data to identify potential brand ambassadors or advocates who could help promote Instagram's initiatives or events?

Answer-

*select distinct u.id as user\_id, t.tag\_name*

*from users u*

*join follows f on u.id = f.followee\_id*

*join photos p on u.id=user\_id*

*join photo\_tags pt on p.id=pt.photo\_id*

*join tags t on t.id=pt.tag\_id*

*where u.id in*

*-- the following code gives users the most number of followers*

*(select user\_id from (select user\_id, follower\_count,*

*dense\_rank() over(order by follower\_count desc) as rnk*

*from (select u.id as user\_id, count(f.follower\_id) as follower\_count*

*from users u*

*join follows f on u.id = f.followee\_id*

*group by u.id) alias1) alias2*

*where rnk=1)*



The above table represents user ids with most number of followers and the tags they have used in the posts they shared on the social media platform. These users can be potential influencers for the topics/tags they used in their posts.

1. How would you approach this problem, if the objective and subjective questions weren't given?

**Answer**- Enough data is not available to answer the questions more meaningfully. For user behaviour, we only have the data of tags which users have liked. Owing to lack of data, few questions could not be answered.

However, if the above questions were not available I would have taken a similar approach. But it is safe to say, these questions have helped me to see the data in a more meaningful way.

1. Assuming there's a "User\_Interactions" table tracking user engagements, how can you update the "Engagement\_Type" column to change all instances of "Like" to "Heart" to align with Instagram's terminology?

Answer

To update the "Engagement\_Type" column in the "User\_Interactions" table, changing all instances of "Like" to "Heart", we can use the following SQL code:

*update User\_Interactions*

*set Engagement\_Type = 'Heart'*

*where Engagement\_Type = 'Like';*

This code will find all rows in the "User\_Interactions" table where the "Engagement\_Type" is "Like" and update them to "Heart".